International Journal of Core Engineering & Management
Volume-6, Issue-08, 2020 ISSN No: 2348-9510

LITERATURE REVIEW ON AUTOMATING THE IT OPERATIONS WITH ANSIBLE

Harika Sanugommula
Harikasanugommula.hs@gmail.com
Independent Researcher

Abstract

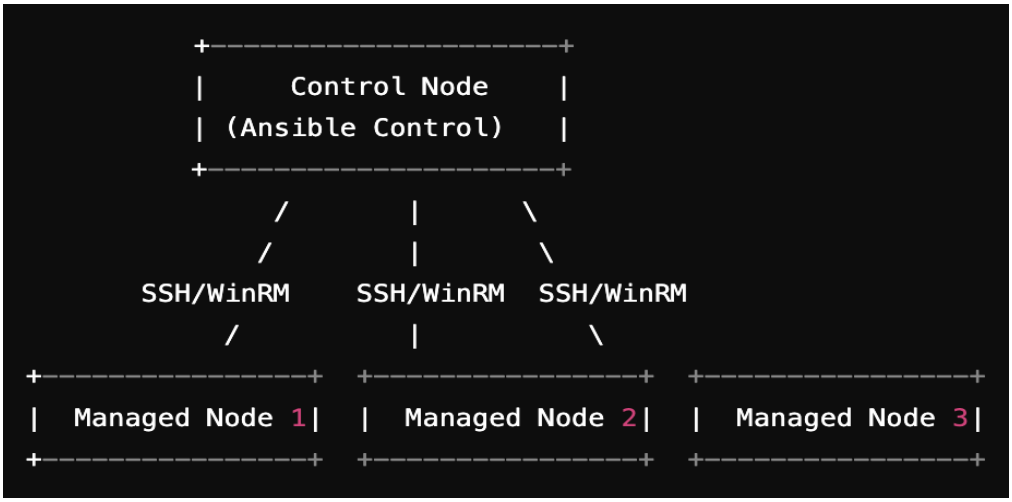
Ansible is an open-source automation tool that simplifies the management of IT infrastructure. It uses a declarative language to define automation tasks, making it accessible for both developers and system administrators. This paper explores Ansible's architecture, components, and its features, supported by diagrams and flowcharts to illustrate its deployment process. We also discuss the advantages and challenges of using Ansible along with real-world use cases that demonstrate its configuration management, application deployment, orchestration.

Keywords : Ansible, automation, configuration management, orchestration, IT infrastructure.

I. INTRODUCTION

As organizations increasingly rely on automated processes for managing their IT infrastructure, tools like Ansible have become essential for achieving operational efficiency. Ansible allows for the automation of repetitive tasks, configuration management, and application deployment using a simple, agentless architecture. This paper provides a comprehensive overview of Ansible, its architecture, key features, and use cases, supplemented by diagrams and flowcharts to illustrate its deployment process.

II. DIAGRAM: ANSIBLE ARCHITECTURE



A simple diagram showing the control node, managed nodes, and communication over SSH/WinRM.

III. ANSIBLE ARCHITECTURE

Ansible operates on a client-server model, where the control machine (server) communicates with managed nodes (clients) over SSH/ WinRM. This agentless architecture simplifies deployment, as no additional software is needed on managed nodes.

Components of Ansible

- **Control Node:** The machine/system where we install Ansible and from which commands are executed
- **Managed Nodes**: The servers or devices that Ansible manages.
- **Inventory:** A file that lists the managed nodes and their details.
- **Modules:** Reusable units of code that perform specific tasks, such as managing files, packages, or services.
- **Playbooks:** YAML files that define the desired state of the managed nodes and describe the tasks to be executed.

Key Features of Ansible

- **Idempotency:** Ensures that tasks can be run multiple times without altering the system state if it is already in the desired state.
- **Declarative Language**: Uses YAML to describe configurations, making it easy to read and write.
- **Extensibility**: Supports custom modules and plugins, allowing users to extend its capabilities.
- **Community Support**: A large community contributes to a rich ecosystem of modules and plugins, enabling users to automate a wide range of tasks.

IV. USE CASES AND EXAMPLES

1. **Configuration Management:** Ansible simplifies server configuration by ensuring consistent setups across multiple environments. For example, a playbook can be used to install and configure a web server.

YAML:

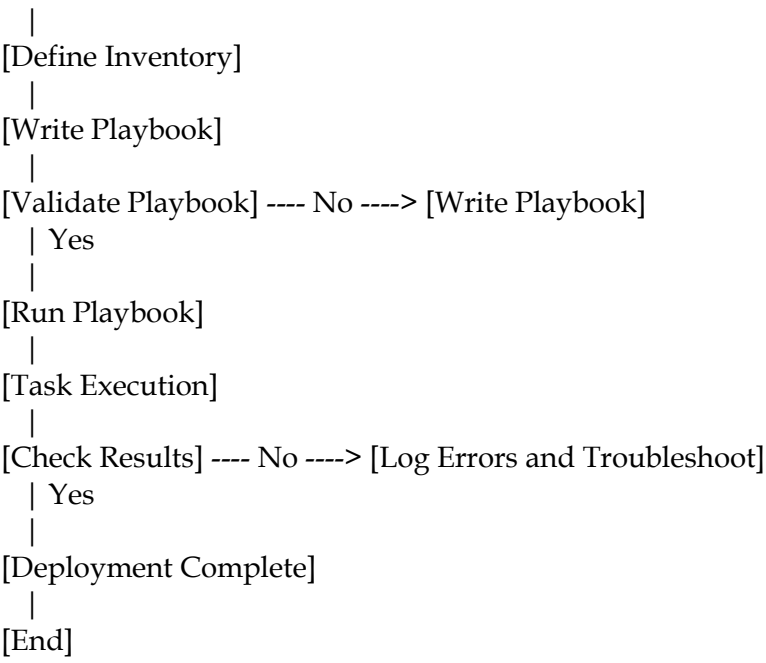
```
- hosts: webservers
  tasks:
    - name: Install Apache
      yum:
        name: httpd
        state: present

    - name: Start Apache
      service:
        name: httpd
        state: started
```

2. **Application Deployment:** Automating application deployment can significantly reduce the time and errors associated with manual processes.

3. **Orchestration:** Ansible can manage multiple services and their dependencies, ensuring that services are started or stopped in the correct order.

Flowchart: Ansible Deployment Process



A flowchart illustrating the steps in the Ansible deployment process, including inventory definition, playbook creation, validation, and execution.

V. ADVANTAGES OF USING ANSIBLE

Ansible offers several advantages, making it a popular choice for automation. One of the key benefits is its simplicity and usability. Ansible uses YAML for playbooks, which is a human-readable language, allowing for easy understanding and implementation of automation scripts. This simplicity results in a low learning curve, enabling users to get started with minimal training and making on boarding faster. Another significant advantage is its agentless architecture.

Ansible works over SSH or WinRM, meaning there is no need to install any agents on managed nodes. This not only simplifies the setup process but also reduces potential security vulnerabilities. Additionally, Ansible is cross-platform compatible, allowing it to manage various systems, including Linux, Windows, and cloud platforms, without requiring specific agents. Ansible ensures consistent state management through idempotency, which means applying the same playbook multiple times will not cause any unintended changes, thereby maintaining stability within the environment. It also boasts an extensive module library, with a wide range of built-in modules that simplify the automation of complex processes. For greater flexibility, users can also create custom modules tailored to specific requirements. Ansible is highly effective for orchestration, with powerful capabilities to manage complex workflows across multiple systems. It

integrates smoothly with CI/CD pipelines, enhancing DevOps workflows.

In terms of scalability, Ansible is efficient for managing large infrastructures, easily handling thousands of nodes. Its dynamic inventory management feature allows for the automatic discovery of hosts in cloud environments. Ansible benefits from a strong community and ecosystem. Its active community contributes extensive documentation, tutorials, and shared resources. Ansible Galaxy, a repository of pre-built roles and collections, helps accelerate development.

Additionally, Ansible's cross-platform management capability makes it a versatile tool that can handle a wide variety of operating systems and cloud platforms, offering centralized automation across diverse environments. It also provides strong security features, utilizing SSH for secure communication with managed nodes, ensuring data protection. Ansible Vault allows for the encryption of sensitive information within playbooks, safeguarding credentials and API keys.

Another notable advantage is cost-effectiveness. Since Ansible is an open-source tool, there are no licensing fees, making it an economical choice for organizations seeking automation solutions.

Real-world use cases of Ansible include configuration management, where it automates server setups to ensure consistent environments, application deployment to streamline the process across multiple servers, and orchestration to manage complex workflows involving numerous systems and services.

VI. CHALLENGES WHEN USING ANSIBLE

Using Ansible for automation and configuration management can come with a few challenges, especially when writing playbooks, roles, and tasks manually. One major issue is dealing with complexity in large playbooks. As the number of tasks and dependencies increases, it becomes harder to maintain and troubleshoot them. Another challenge is ensuring that tasks are idempotent, meaning they can be safely run multiple times without causing problems. If tasks aren't idempotent, you might run into issues like configuration drift. Debugging errors can also be tricky because Ansible's error messages can sometimes be unclear, making it harder to figure out what went wrong.

Managing sensitive information, like passwords and API keys, securely is another hurdle. While Ansible Vault helps with this, it requires careful handling to avoid potential security risks. In addition, ensuring that playbooks work consistently across different environments can be difficult. For example, differences in operating systems or package managers can cause tasks to fail. Testing playbooks can also be time-consuming, and without proper testing, issues might not show up until they're in production.

Managing dependencies and roles can add complexity, especially in large environments. Poorly designed roles or incorrect dependencies can cause failures during execution. Performance can also be an issue, particularly when working with many hosts, as Ansible relies on SSH, which can slow things down. For newcomers, the learning curve can be steep, and it might take time to get comfortable with the tool. Collaboration can also be challenging when multiple team members are

working on the same playbooks, especially without proper version control, which can lead to conflicts.

Another challenge is that Ansible doesn't have a built-in graphical user interface (GUI), which may be a downside for users who prefer working with one. Managing changes across multiple environments and keeping systems consistent can be difficult without good version control. Lastly, Ansible doesn't offer an easy way to roll back changes if something goes wrong, which can leave systems in an inconsistent state.

VII. CONCLUSION

Ansible is a powerful automation tool that enhances efficiency and consistency in IT operations. Its agentless architecture, combined with features like idempotency, extensibility, and community support, makes it a valuable asset for organizations seeking to streamline their infrastructure management. As automation becomes increasingly important, Ansible will continue to play a crucial role in driving operational excellence.

REFERENCES

1. A. Moser, S. Ostermann, and R. Prodan, "Evaluating the Benefits of the Ansible Automation Tool for Cloud-Computing Workflows," in Proc. 6th IEEE Int. Conf. on Cloud Computing Technology and Science (CloudCom), 2014, pp. 342-347.
2. Limoncelli, T. A., Chalup, S. R., & Hogan, J. The Site Reliability Workbook: Practical Ways to Implement SRE. O'Reilly Media. 2016.
3. Hutter, A. Ansible for DevOps: Server and Configuration Management for Humans. Leanpub 2018.
4. S. Sharma, "Managing Infrastructure with Ansible," Proceedings of the 2016 International Conference on Cloud Computing and Virtualization, San Francisco, CA, 2016, pp. 45-50.